基于特征选择的虚拟化系统语义鸿沟桥接研究 *

娄 睿[1]，蒋烈辉[1, 2]

(1. 信息工程大学 四院，郑州 450002; 2. 数学工程与先进计算国家重点实验室，郑州 450002)

摘 要：虚拟化系统的强隔离性质在为安全机制部署提供可靠环境的同时，也引入了语义鸿沟问题。针对现有研究普遍依赖的软件体系结构信息、数据结构和控制流容易被篡改，采用的检测算法在客户机状态识别方面效率较低等问题，设计了特征构造和窗口标记的方法对虚拟机数据进行预处理，以满足实施数据挖掘的必要条件，建立了基于特征选择的虚拟化系统语义鸿沟桥接模型，能够仅依赖硬件体系结构数据构建虚拟机执行模式并进行安全检测。实验结果表明，所设计的系统模型能够筛选出关键的虚拟机特征，并有效地识别出客户机异常行为，提高语义鸿沟的桥接效率，为处理语义鸿沟问题提供了一种可行方案。

关键词：虚拟化安全；语义鸿沟；机器学习；特征构造；特征选择；异常检测

中图分类号：TP309 doi: 10.3969/j.issn.1001-3695.2017.12.0761

Research on bridging semantic gap in virtualization system based on feature selection

Lou Rui[1], Jiang Liehui[1, 2]

(1. *Fourth Department*, *Information Engineering University*, *Zhengzhou* 450002, *China*; 2. *State Key Laboratory of Mathematical Engineering & Advanced Computing*, *Zhengzhou* 450002, *China*)

Abstract: The strong isolation property of the virtualization syste introduces the semantic gap problem while providing a reliable environment for the deploying the security mechanism. Current research generally relies on the information of software architecture which is not reliable, for the data structures and control flows are easy to be illegally manipulated. And the detection algorithm employed in related research has the low efficiency in identification of guest state. For these problems, this paper designed the feature construction and window mark to preprocess the captured data so as to meet the necessary conditions of carrying out data mining, and then established the semantic gap bridging model of virtualization system based on feature selection, which can build the execution mode of virtual machine and carry out the security detection only relying on the hardware architecture data. Test results show that proposed model can screen out the key features of virtual machine and effectively identify the abnormal behavior of guest system, which lead to the efficiency improvement of bridging semantic gap. This scheme provides a feasible solution for dealing with the problem of semantic gap.

Key Words: virtualization security; semantic gap; machine learning; feature construction; feature selection; anomaly detection

0 引 言

随着信息技术的发展计算系统日趋复杂并暴露出更多的攻击面，恶意程序在攻击方式上呈现多样化的同时也逐步向更高特权级渗透，给系统安全防护带了巨大挑战。虚拟化技术凭借其底层控制、域间隔离、域外监控的特性，在安全领域得到了广泛应用。虚拟化系统的强隔离性质在为安全部件提供可靠环境的同时，也导致虚拟机管理器(virtual machine monitor, VMM)视图和客户机视图之间存在语义鸿沟，使安全部件难以准确获取客户机内部状态，制约了安全机制的有效发挥。语义鸿沟问题已成为目前虚拟化安全研究的一个热点。

为了解决语义鸿沟问题，Garfinkel 等人首次提出了虚拟机自省(Virtual Machine Introspection, VMI)的概念[1]。VMI 是指在虚拟机外部，通常是从 VMM 或其他特权系统的有利角度监控、获取客户机原始数据，并借助特殊服务程序从中提取、还原客户机内部状态和事件的过程。Garfinkel 等人设计了首个 VMI 原型系统 Livewire，基于 Linux crash 分析工具建立了操作系统接口库，该接口库主要用于将 VMM 提供的虚拟机状态转换成操作系统级视图。InSight[2]主要依据内核源码与符号表获得 VMM 视图与客户机视图之间的映射关系进而还原出系统相应信息。KDD[3]对内核源码进行静态指向分析以消除指针关系的歧义性，从而构建出精确的内核数据定义，能够精确映射虚拟机物理内

基金项目：国家自然科学基金资助项目（61572516）；河南省科技攻关项目（162102210033）

作者简介：娄睿（1989-），男，河南新乡人，博士研究生，主要研究方向为操作系统、虚拟化安全（vivagin@yeah.net）；蒋烈辉（1967-），男，教授，博士，主要研究方向为逆向工程、大数据处理.

存并提取出运行时对象以桥接语义鸿沟。RTKDSM[4]专用于对客户机状态的实时监控和分析来处理语义鸿沟问题。ODinn[5]通过预先缓存重要信息，解析内核源码并扫描完整内存结构来还原目标系统状态。IVirt[6]通过地址转换和内容定位获得虚拟机所需的内存数据，进而验证客户应用程序的完整性以判断目标系统状态。VMST[7]通过自动识别和重定向自省数据到内核空间的方式重构高级操作视图。

虚拟化与机器学习结合是当前虚拟化研究的新方法，这类方法通常借助虚拟化提取客户机数据流，并采用数据挖掘、统计学、模式识别等方法建立目标系统的执行模式，在此基础上进行虚拟机性能评估、负载特征分析、虚拟机状态检测等。Anand 等人在虚拟机网络入侵检测中采用了基因算法进行特征选择，并采用了模糊 SVM 算法对审计数据进行分类[8]。Udaya 等人采用了神经网络算法检测虚拟域中的网络包[9]。Xu 等人通过引入机器学习对虚拟内存访问模式进行监视和分类进行恶意程序检测[10]。Mishra 等人对所收集的信息应用决策树进行行为学习并分类，进而检测恶意系统调用模式[11]。

尽管上述方法在一定程度上修复了语义鸿沟，提升了操作系统安全性，但这些方法普遍依赖的软件体系结构信息存在数据结构和控制流容易被窜改的缺陷，同时所采用的检测算法在客户机状态识别方面效率较低。尽管硬件体系结构信息能够提供准确的系统状态，但现有方法并不能从中推导出必需的客户机状态，也无法有效识别出客户机中的恶意行为。

本文基于前期工作[12]进行拓展研究，作为一种基于虚拟化的防护系统，协作型虚拟化安全模型(cooperating virtualization security model, CVSM)具备良好的抗攻击性，但同时也在系统低级状态和系统行为之间引入了语义鸿沟，使 VMM 很难准确获取客户机内部状态。从 CVSM 的虚拟化层能够提取出丰富的硬件体系结构信息，基于这些信息可以构造出多样的虚拟机特征。本文针对 CVSM 进行适应性设计，使之满足进行数据挖掘的必要条件，建立了基于特征选择的语义鸿沟桥接模型，通过特征选择处理虚拟机数据，并以分类算法进行安全检测，用于提高客户机中异常行为的识别能力。

1 虚拟化层数据分析

根据硬件体系结构数据难以被窜改和旁绕的特性，以硬件体系结构层（或称虚拟化层）收集的信息作为数据源，在该层次能够访问和利用软件层无法观测到的接口，如段页保护、中断和异常处理、任务管理、cache 管理、虚拟机扩展等。在本文研究中，CVSM 中使用的数据来源于以下几个方面：①虚拟机状态，如 CR3、IDTR、GDTR、LDTR、I/O 等；②虚拟机扩展，如上下文切换陷入、特权指令陷入、VMCALL 指令陷入等；③中断或异常，如页故障、通用保护异常等。

虚拟化系统中恶意行为高低级语义之间存在一定的对应关系[13]，表 1 列出了这种映射关系。借助高低级语义映射可以在更低层次提取与客户机安全相关的信息，并从中状态推导出客户机内部状态。

表 1 恶意行为的高低级语义映射

高级语义	低级语义
中断钩子	IDTR 与系统调用 MSR 操作，系统中断陷入，用户中断陷入，页故障异常，通用保护异常
进程隐藏	上下文切换陷入，CR3 写操作
模块隐藏	用户中断陷入，页故障异常，通用保护异常
网络活动隐藏	虚拟 I/O 读写
虚拟机 Rootkit	虚拟机进入陷入
网络攻击	虚拟 I/O 读写
恶意进程	虚拟 I/O 读写，用户中断陷入，页故障异常，通用保护异常，无效操作码异常，调试异常

CVSM 适用于两种场景：①用于入侵检测，对正常的工作负载提供安全防护；②用于安全测试或蜜罐诱捕，收集、分析恶意程序的攻击行为。无论是入侵检测还是安全测试 CVSM 都需要在虚拟化层收集大量数据，为达到数据收集目的系统会设置并产生频繁的虚拟机退出操作，每个数据集都可能包含上千个原始数据，这些数据通常具有高维、类不平衡、类基本平衡三个特点，不仅增加了计算时间也降低了检测精度，因此在实验过程中并不适合对这些数据进行直接处理。

2 CVSM 中的语义鸿沟桥接

2.1 语义鸿沟桥接模型

从虚拟化层收集的数据具有高维、冗余、包含噪声等特点，并不直接适用于学习算法。对原始数据进行预处理、为机器学习过程提供准确而简洁的高质量数据，是提高学习算法效率的重要环节。典型的机器学习过程如图 1 所示，其中学习环节包含在数据训练阶段。

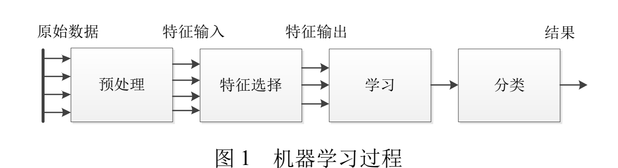


图 1 机器学习过程

由图 1 可知，学习算法通常并不直接对原始数据进行处理，为此需要对 CVSM 进行适应性设计，对所收集的数据进行预处理以实施特征选择，进而筛选出符合学习算法的有效输入。数

据输入到学习算法之前的必要处理包括：

a）数据收集。CVSM 必须能够持续收集数据，使学习算法能够从中提取充足的特征并建立虚拟机执行模式。所收集的数据中应该包含正常数据和异常数据，便于对学习算法进行训练和测试。

b）数据描述。CVSM 应该对所收集的数据进行描述，将数据转换成可以度量与比较的数值（或特征）有助于学习算法识别离群点。数据包括属性描述（二元属性、数值属性等）和统计描述（距离、方差等）。

c）数据标记。CVSM 应该具备数据标记能力，采用带标记的正常数据和异常数据对学习算法（监督式学习）进行训练。由于 CVSM 中存在语义鸿沟，实现数据标记并不容易。

d）在线和离线数据处理。学习算法训练阶段 VMM 往往需要统计某个时间段内的数据量与分布，要求 CVSM 提供离线缓存功能。检测阶段则要求 CVSM 能够及时处理数据，提高其响应能力并减少空间存储。因此 CVSM 应该同时具备在线和离线数据处理功能。

当前的入侵检测技术主要有误用检测和异常检测，CVSM 的一个重要设计目标是能够对未知系统状态进行判断，因此本文采用了异常检测方法，构建了基于特征选择的 CVSM 语义鸿沟桥接模型（如图 2 所示），包括客户机数据收集、客户机数据筛选、客户机状态检测三个阶段。

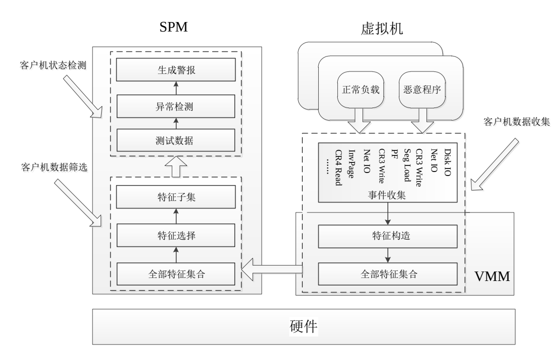


图 2　基于特征选择的语义鸿沟桥接模型

2.2　数据处理流程

根据数据处理部件的位置差异，可以将机器学习在 CVSM 中的应用模型分为前端和后端，如图 3 所示。

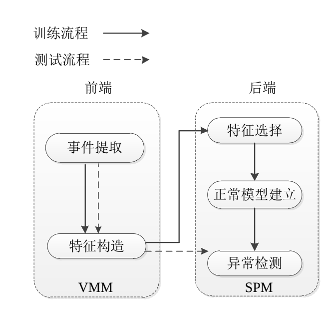


图 3　机器学习模型组成及数据流程

前端在 VMM 中，包括事件提取和特征构造两个部分：a)事件提取,VMM 捕获硬件体系结构层产生的原始数据，如磁盘和网络 I/O 操作、页故障、TLB 刷新等；b)特征构造,借助统计学方法将原始数据转换成相应的特征，用于后续的机器学习。后端在 SPM 中，包括特征选择、正常模式建立和异常检测三个部分:a)特征选择,从全部特征集合中筛选出少数高质量特征，降低时间复杂度并提高学习算法效率；b)正常模式建立：构造出虚拟机负载正常的执行模式；c)异常检测,识别偏离正常执行模式的异常行为。

由图 3 可知，机器学习在 CVSM 上的数据流程包括训练阶段和测试阶段。

a)训练阶段。首先运行正常负载，在前端收集事件并构造出尽可能多的特征，然后将这些特征递交到后端进行处理，在筛选特征的基础上建立正常执行模式，之后同时运行包含正常和异常行为的负载，评估这些行为与正常行为的偏离程度并作为分类预测的依据。

b)测试阶段。在前端运行任意类型的负载并收集事件，然后仅构造经过筛选的高质量特征，后端的异常检测单元将特征与训练阶段建立的正常执行模式对比，确定事件中是否包含异常行为。

3　虚拟机特征构造

3.1　事件提取

CVSM 中事件是指从硬件体系结构层提取的原始数据和信息。VMM 对涉及体系结构的事件进行捕获并重新解释以确保多个虚拟机的正常运行，是虚拟化中基本的“限权”原则。典型事件包括执行特权指令，访问共享资源（内存、外设等）。事件所含信息的丰富程度与 CVSM 的性能密切相关，为了能够更准确地还原出客户机内部状态，必须广泛收集各种类型的事件。事件主要包括两类：

a）虚拟机事件。与虚拟机中客户机相关的体系结构或系统级事件，如客户机修改控制寄存器、刷新 TLB、写磁盘等。

b)VMM 事件。从 VMM 提取的与其自身状态相关的事件。客户机的状态以及客户机与 VMM 之间的交互也会影响到 VMM 状态，例如客户机执行特权指令时会从虚拟机地址空间陷入 VMM 地址空间。

除了监控事件的产生外还需要提取出与之相关的信息，例如对于磁盘 I/O 事件需要给出目标扇区、字节大小和读写类型等具体内容。表 2、3 分别列出了所提取的虚拟机事件和 VMM 事件，所提取事件应该有助于预测客户机状态改变，因此同时给出了与这些事件相对应的客户机高级语义。

3.2　事件流分割

TCP 协议中采用了窗口机制进行流量控制，本文也采用了窗口概念对事件流进行分割，以提高后端学习算法的数据处理效率。基本方法为：首先将事件流分成等时间的连续片段，之后借助统计学方法将片段中的事件转换成特征值，最后将窗口（包含一个片段中所有的特征值）发送到后端分析。窗口如图 4 所示。

表 2　虚拟机事件以及客户机中可能的高级语义

虚拟机事件	事件内容	高级语义
磁盘 I/O	磁盘扇区，字节大小，读或写	磁盘读/写事件
网络 I/O	字节大小，发送或接收	网络发送/接收事件
读/写 CR	寄存器号，数值，读或写	与具体寄存器相关，例如写 CR3 可能为进程切换
页故障	错误号，EIP 数值	违反写保护或启动新进程
TLB 刷新	全局标记，新 CR3	上下文切换
页无效	页无效的线性地址	程序试图越界访问
加载段描述符	段寄存器，基地址等	程序启动
当前特权级	特权级	当前运行用户或内核代码
LDT，GDT，TSS 访问	CR2 中的故障地址	程序启动或上下文切换
模式指定寄存器	寄存器，数值，读或写	内核设定 CPU 工作环境，标志 CPU 工作状态
快速系统调用	SYSENTER_CS_MSR	用户到内核的快速调用

表 3　VMM 事件以及客户机中可能的高级语义

VMM 事件	事件内容	高级语义
设置或退出虚拟化模式	—	进入或退出支持客户机运行的模式
启动虚拟机	虚拟机控制结构地址	客户机开始或重新运行
读写虚拟机控制结构	虚拟机控制结构中相应的子结构地址	获取客户机当前状态，设置或修改客户机的执行模式
虚拟机调用	—	客户机预知运行于虚拟机并向 VMM 发出退出请求
设置虚拟机退出处理程序	退出程序入口及内容	客户机对真实机的操作能力被限制、改变

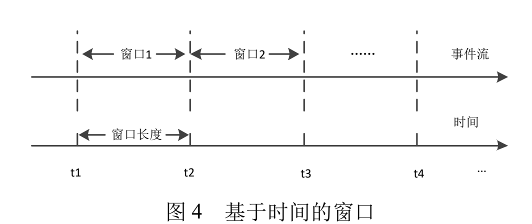


图 4　基于时间的窗口

采用基于时间的窗口有两个优势：①每个窗口代表了相等的执行时间，方便窗口之间进行比较；②能够以窗口为单位进行属性判断（正常或异常）。窗口长度也会对系统性能造成影响：较长的窗口可以捕获更多行为，较短的窗口则可以缩短检测时间。实验发现窗口长度在 2~4 s 时可以取得较好的平衡，相应的窗口不仅包含了丰富的信息能够完成分类，也能在较短时间内识别出异常行为。

3.3　特征设计空间

CVSM 中前端向后端提供的特征包括原始特征和处理特征。原始特征包含所有从事件中提取的未处理信息。原始特征之间可能非常相近而呈现冗余，例如 CR3 修改通常伴随 TLB 刷新操作，原始特征中可能只有部分信息与系统状态改变相关，如果采用原始特征作为输入，学习算法必须处理上述问题；处理特征是对事件进行过滤、聚合或变换得到的特征，并不包含从事件中提取的所有信息而是信息的组合模式。本文采用了处理特征作为学习算法的输入。

原始事件能够用于特征构造的特征维包含多个方面:a)事件类型,基于单一事件构造的特征可以从特定角度表达系统行为,基于多个事件构造的特征则能够从不同角度表达系统行为；b)事件信息,既可以根据事件是否发生构造特征(如产生 CR3 写操作)，也可以根据事件所附带的信息构造特征（如 CR3 写操作的具体数值)；c)时间,包括客户机所感知的虚拟时间和 VMM 所感知的真实时间,可以在有限时间内构造出包含特征的窗口,基于时间信息可以生成一系列连续的窗口，时间也可以用于统计截止到某个时间点的事件累积状态；d)事件语义,不同抽象级所代表的语义能够从不同角度表达系统行为，因此也可以从语义角度构造特征。虚拟机特征反映了客户机行为，可以构造出基于系统和进程的特征；VMM 特征既可以反映 VMM 行为，也能间接反映客户机行为。

根据上述几种特征维，本文采用了统计学方法将事件转换成特征选择算法易于处理的特征向量,构造了两类虚拟机特征：速率特征和关系特征。

3.3.1 速率特征构造

速率特征描述了一个窗口内特定事件的产生频率，通过存储该窗口长度内的事件流并统计其中每类事件发生的次数实现，图 5 表示了从事件流中构造出速率特征的实例。所构造的速率特征包括页故障、控制寄存器修改、磁盘和网络的 I/O 操作等，这些特征为预测系统行为提供了丰富信息。

表 4 列出了所构造的部分速率特征，其中以 RATE-VM 开头的表示虚拟机事件，以 RATE-VMM 开头的表示 VMM 事件。

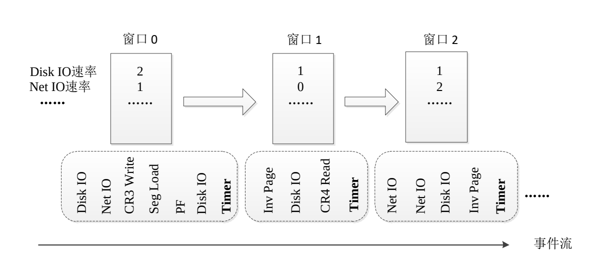


图 5　速率特征构造

表 4　部分速率特征实例

特征名称
RATE-VM-PAGE-FAULT-P-0
RATE-VM-TLB-FLUSH
RATE-VM-GDT-WRITE
RATE-VM-TSS-WRITE
RATE-VM-CPL-SET
RATE-VM-CR0-WRITE
RATE-VM-CR3-WRITE
RATE-VM-TRAP-RATE
RATE-VM-PAGE-FAULT
RATE-VMM-SET-GUEST-EXIT-HANDLER
RATE-VMM- RESUME-VM-EXECUTION
……

3.3.2 关系特征构造

不同窗口中某些事件可能具有相同的速率但产生次序不同，由于速率特征只能反映单个特征的发生频率，并不能反映事件之间的关系，因此速率特征认为这些窗口相同。为解决上述问题，可以将事件的产生和事件的统计值相结合构造关系特征，图 6 表示了这类特征的构造方法，X 轴表示事件 A、B 的产生次序，Y 轴表示 A、B 产生次数的差异，如果事件 A 产生则 Y 值加 1，如果事件 B 产生则 Y 值减 1。

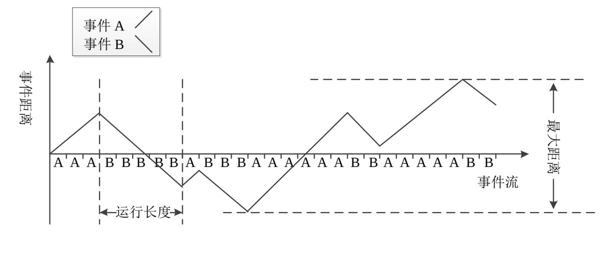


图 6　基于事件产生和统计值的关系特征

CVSM 中所构造的关系特征包括以下数值：a) 序列总数(total sequence),所有相同类型事件连续产生的总次数；b) 最大差异(max discrepancy),事件 A、B 在 Y 轴方向上的最大差值；c) 最大运行长度(max run length),相同事件连续产生次数的最大值；d) 最小运行长度(min run length),相同事件连续产生次数的最小值；e) 平均运行长度(mean run length),所有运行长度的平均值；f) 曼-惠特尼 U 检验(Mann-Whitney U test),用于度量事件 A、B 交叉产生时的随机程度；g) 曼-惠特尼 U-P 值(Mann-Whitney U-P value)：表示曼-惠特尼 U 值的规范化统计，目的是方便对特征值进行比较。

表 5　部分关系特征实例

特征名称
REL-VM-GDT-WRITE-VS-CPL-SET-0[maxDiscrepancy]
REL-VM-DISKIO-READ-VS-DISKIO-WRITE[maxDiscrepancy]
REL-VM-DISKIO-READ-VS-NETWORKIO-WRITE[MannWhitneyU]
REL-VM-DISKIO-READ-VS-NETWORKIO-WRITE[MannWhitneyP]
REL-VMM-ENTER-VM-MODE-VS-EXIT-VM-MODE[MannWhitneyP]
REL-VM-CR3-WRITE-VS-TSS-WRITE[meanRunValue]
REL-VM-TSS-WRITE-VS-TLB-FLUSH[maxDiscrepancy]
REL-VM-PAGE-FAULT-P-0-VS-PAGE-FAULT-P-1[MannWhitneyP]
REL-VM-PAGE-FAULT-ID-0-VS-PAGE-FAULT-ID-1[maxDiscrepancy]
……

采用上述方法在 CVSM 中构造了关系特征，数据源包括：网络 I/O，磁盘读写字节数，内存读写操作引发的页故障等。表 5 列出了所构造的部分关系特征实例，其中 REL-VM 开头的表示虚拟机事件之间的关系特征，REL-VMM-VM 表示 VMM 和虚拟机事件之间的关系特征，REL-VMM 表示 VMM 事件之间的关系特征。

4　窗口标记

测试负载生成的正常窗口与恶意程序生成的异常窗口构成了窗口样本，本文首先采用了 Boosting 类算法完成窗口样本的特征选择，再借助分类算法识别异常窗口。训练 Boosting 类算法时需要对样本进行标记，因此 CVSM 需要提供窗口标记机制，用带标记的窗口训练特征选择算法。

对窗口进行标记存在两方面困难：①关于“恶意行为”并无准确定义；②即使可以借助客户机内部的安全软件识别出恶意行为，但语义鸿沟导致 VMM 很难判断原始数据中是否包含恶意行为。为桥接语义鸿沟，需要在客户机的高级操作和虚拟机数据之间建立关联。需要说明的是，事件流以窗口为单位递交给后端的学习算法，即使建立了高低级语义关联，学习算法对事件的识别粒度也只能精确到窗口级别。对窗口进行标记的关键在于使 VMM 能够确定恶意行为活动周期，并将活动周期内所生成的窗口标记为异常窗口。

由于系统行为的高低级语义之间存在映射关系，VMM 可以根据低级事件的活动规律预测出客户机高级行为状态，从而直接对窗口进行标记。由于正常负载在运行期间不会或极少创建新的进程，如果在某个时间段内进程数量显著增加，则这些

进程很可能是恶意程序。VMM 需要区分并跟踪恶意程序产生的事件，将包含这些事件的窗口标记为异常窗口。

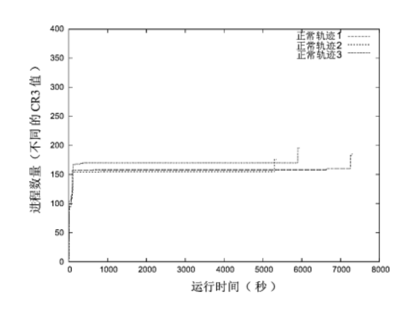


图 7　正常负载运行中的进程增长

进程具有不同的页目录基地址，该地址可以通过 CR3 获取，因此 VMM 通过收集、识别 CR3 可以区分客户机中的进程。图 7 反映了负载 Database、Web 和 EMail 运行过程中 VMM 所观测到的进程增长情况，从中可以看出在负载的主要运行区间内进程创建的数量并无显著增加。为比较注入恶意程序后客户机中的进程增长数量，在负载运行期间启动了四个恶意程序，相应的进程增长情况如图 8 所示，从中可以看出正常负载和恶意程序交替运行过程中存在明显的进程增长现象，新创建的进程与恶意程序有关。

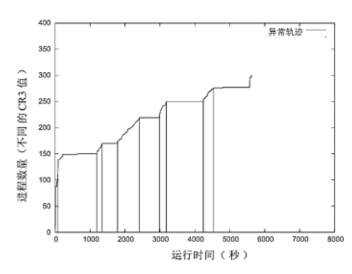


图 8　正常负载与恶意程序混合运行中的进程增长

基于以上观测和分析，VMM 窗口标记的步骤如下：①VMM 缓存两个窗口并统计其中的 CR3 数值，将 CR3 数量显著变化区间内的 CR3 定义为异常 CR3 并作记录；②检测所缓存的窗口中是否包含异常 CR3，如果包含则说明该窗口含有恶意进程所产生的事件，将该窗口标记为异常窗口；③两个窗口长度内会产生大量的 CR3 操作，从一次异常 CR3 产生直至切换到正常 CR3 之间的所有事件都由同一进程产生，因此包含这些事件的连续窗口都是异常窗口。窗口标记的示意图如图 9 所示，其中灰色部分表示异常 CR3。

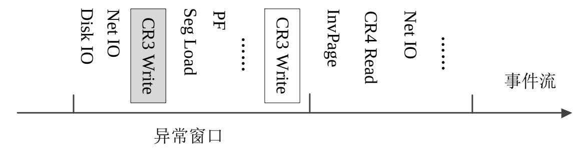


图 9　窗口标记示意图

5　测试与分析

本文测试的主要目的是评估 CVSM 从硬件体系结构事件中推导和识别客户机状态的能力，测试应该体现：①对反映系统状态的关键虚拟机特征的筛选；②恶意行为的检测效果；③性能开销。

5.1　测试集

CVSM 的一个主要优势在于方便部署，向其他平台移植时只需要修改与体系结构相关的事件收集方式，无须对客户机操作系统进行修改，因此可以支持多种类型的虚拟机。典型的服务器应用通常包括一个主进程和一个后台进程，这类负载所产生的系统行为相对稳定，因此本文针对虚拟服务器平台进行了测试。为产生多种类型的负载（CPU 密集型、磁盘 I/O 型、网络 I/O 型），反映多样的系统行为，采用了不同类型的服务器和虚拟应用。用于产生正常负载的三种测试集如表 6 所示。

表 6　产生正常负载的测试集

数据集	服务器	虚拟应用	负载
Database	数据库管理系统	MySQL Server	在线交易处理基准(TPC-C)
Web	网络服务器	Apache HTTP Server	Apache 带宽基准(ab)
EMail	邮件服务器	Exchange Server	Exchange 负载模拟器(LoadSim)

同时为在服务器上生成异常负载，从 Malfease、Open Malware、VX heavens、VXShare 等库中收集了已知和未知的恶意软件。根据行为差异将这些软件分为四类：Trojan、Infostealer、Downloader 和 Backdoor，可以产生修改数据、安装程序、泄露信息、拒绝服务等恶意行为。

5.2　虚拟机特征筛选

虚拟机特征选择的目的是从 VMM 层所收集的事件中提取出对识别异常行为有价值的特征。特征选择算法采用 AdaBoost-CR[14]，该算法采用贡献率(Contribution Rate, CR)作为权重来衡量每个特征的重要程度，通过对特征权重排序可以对虚拟机特征进行有效分析。

5.2.1 特征关联度

关系特征反映了不同事件之间的交互，在增强学习算法输入的同时也产生了冗余特征，冗余特征之间强相关且不包含太多额外信息。图 10(a)表示了冗余特征“RATE-VM-CR3-WRITE”和“RATE-VM-TSS-WRITE”之间的分布。但相关并不一定意味着缺乏信息，存在噪声的情况下同时选择相关的特征有助于完成更好的分类及噪声消除，图 10(b)表示大致冗余特征“RATE-VM-CR0-WRITE”和“RATE-VM-CR4-WRITE”之间的分布。

如果两个类的条件密度的协方差矩阵在第一主方向相同，但对类中心进行相同的移位，则这样的特征虽然相关但并不冗余，同时选择这些特征能够提供有利于分类的信息。图 10(c)表示了类条件依赖特征“RATE-VM-TLB-FLUSH”和“RATE-VM-INVALIDATE-PAGE”之间的分布。尽管根据事件交互构造了关系特征，但并非所有事件交互都有意义，有可能产生噪声和无

关特征。无关特征间具有相同的类条件概率密度函数，并不包含有价值的信息，实验表明与加载段描述符有关的事件之间产生的是无关特征，图 10(d)表示了无关特征的分布。

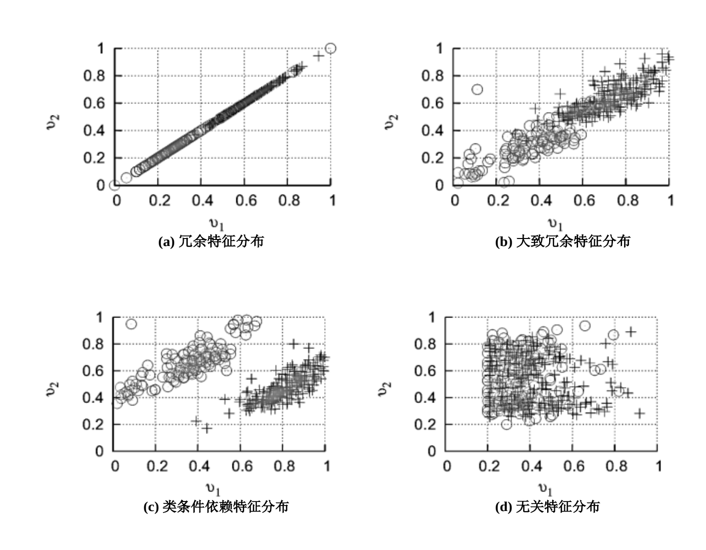


图 10　虚拟机特征关联度与分布

5.2.2 频繁特征

采用 AdaBoost-CR 算法进行虚拟机特征选择，可以为每一种负载选择出带 CR 权重的特征，表 7(a)(b)(c)分别列出了 Database、Web 和 EMail 上选出的权重最高的 5 个虚拟机特征。可以看出，①关系特征所占比重大于速率特征，三个测试负载中所有的速率特征比重小于 20%，而且关系特征中也包含了一部分速率特征所描述的信息，表明特征选择算法能够去除冗余特征。②不同负载上所选的特征具有相似性，三个测试负载中权重较高的特征分为五种类型：GDT 特征、TLB 刷新特征（包含 CR3 操作事件）、磁盘与网络 I/O 特征、页故障特征、VMM 特征（包含模式切换和页故障）。

Databse 负载上最好的特征为"REL-VM-PAGE-FAULT-P-0-VS-PAGE-FAULT-P-1[MannWhitneyP1]"，这是因为 Database 为磁盘密集型负载会产生大量的磁盘读写操作，数据在磁盘和内存之间频繁交换会导致大量的页故障事件。Web 负载上最好的特征为"REL-VM-NETWORKIO-READ-VS-DISKIO-WRITE [MannWhitneyP1]"，是由网络 I/O 事件和磁盘 I/O 事件组成的关系特征，此类事件也与 Web 的负载类型一致。Email 负载中最好的特征为"REL-VMM-VM-EXIT-VM-MODE-VS-PAGE-FAULT-US-0[MannWhitneyP1]"，该特征反映两种事件的关系：①系统从虚拟机模式退出到 VMM 模式，由 VMM 对客户机敏感操作进行解释。②VMM 在解释客户机行为中产生了页故障。

表 7　负载上 5 个权重最高的特征

(a) Database 负载

特征名称	CR 权重
REL-VM-PAGE-FAULT-P-0-VS-PAGE-FAULT-P-1[MannWhitneyP1]	0.376832
REL-VM-NETWORKIO-READ -VS-DISKIO-WRITE [MannWhitneyP2]	0.120571
REL-VM-CR3-WRITE-VS-TSS-WRITE[meanRunLength]	0.111837
REL-VMM-VM -EXIT-VM- MODE-VS-PAGE-FAULT-US-0[MannWhitneyU1]	0.109041
REL-VM-GDT-WRITE-VS-CPL-SET-3[MannWhitneyU1]	0.099245

(b) Web 负载

特征名称	CR 权重
REL-VM-NETWORKIO-READ -VS-DISKIO-WRITE [MannWhitneyP1]	0.456714
RATE-VM-DISKIO-READ	0.265052
REL-VM-CR3-WRITE-VS-CR4-WRITE[meanRunLength]	0.166301
REL-VM-CR3-WRITE-VS-CR0-WRITE[MannWhitneyP2]	0.105268
RATE-VM-PAGE-FAULT-WR-1	0.074971

(c) EMail 负载

特征名称	CR 权重
REL-VMM-VM-EXIT-VM-MODE-VS-PAGE-FAULT-US-0[MannWhitneyP1]	0.334201
REL-VM-PAGE-FAULT-P-0-VS-PAGE-FAULT-P-1[MannWhitneyP1]	0.218797
RATE-VM-DISKIO-WRITE	0.144215
RATE-VM-NETIO-WRITE	0.112438
REL-VM-TSS-WRITE-VS-TLB-FLUSH[maxRunLength]	0.059907

对虚拟机的特征分析表明 CVSM 能够有效筛选出高质量特征，也说明并非所有的原始事件都能够产生有价值的特征，仅需要构造与高频度事件相应的特征，有利于提高 CVSM 检测率和检测速度。

5.3　异常检测分析

在特征选择环节后需要采用分类算法，根据所选特征子集

对 CVSM 产生的窗口数据进行分类。K-NN[15]是一种经典的懒惰学习算法，通过计算新数据与训练数据特征值之间的距离区分近邻点以实施分类；LOF[16]是基于密度的离群点检测方法中的代表性算法，能为数据集中的每个点计算一个离群因子 LOF 值以衡量该点是否异常。考虑到 K-NN 和 LOF 两种分类算法在数据处理方式上具有显著差异，并且在主要指标方面具有较强的互补性，因此选择了 K-NN 和 LOF 两种算法对所选特征子集进行分类。表 8 和表 9 分别给出了特征选择算法与 K-NN、LOF 算法组合时三种负载的检测率、误报率以及 AUC 结果。

表 8　特征选择算法和 K-NN 算法组合时的测试结果

	Database			Web			Email		
恶意程序	FP(%)	TP(%)	AUC(%)	FP(%)	TP(%)	AUC(%)	FP(%)	TP(%)	AUC(%)
Downloader	9.00	95.0	86.0	4.00	97.0	93.0	3.00	96.0	93.0
Infostealer	6.00	98.0	92.0	5.00	98.0	93.0	6.00	98.0	92.0
Trojan	8.00	93.0	85.0	2.00	98.0	96.0	4.00	92.0	88.0
Backdoor	4.00	94.0	90.0	2.00	96.0	94.0	2.00	98.0	96.0
平均值	7.00	95.0	88.0	3.00	97.0	94.0	4.00	96.0	92.0

表 9 特征选择算法和 LOF 算法组合时的测试结果

	Database			Web			Email		
恶意程序	FP(%)	TP(%)	AUC(%)	FP(%)	TP(%)	AUC(%)	FP(%)	TP(%)	AUC(%)
Downloader	8.00	93.0	85.0	6.0	96.0	90.0	6.00	90.0	84.0
Infostealer	5.00	96.0	91.0	5.0	97.0	93.0	4.00	91.0	87.0
Trojan	7.00	93.0	86.0	3.00	98.0	95.0	8.00	89.0	81.0
Backdoor	8.00	95.0	87.0	3.70	94.0	91.0	8.00	92.0	84.0
平均值	7.00	94.0	87.0	4.30	96.0	92.0	7.00	91.0	84.0

为直观展示 CVSM 异常检测效果，图 11 展示了特征选择算法和 LOF 组合时在 Downloader 上的检测结果，其中 Y 轴表示分类结果的可信度。可以看出当 Downloader 运行后，相对于正常窗口，异常窗口具有更高的分类可信度，进而表现出更好离群性。

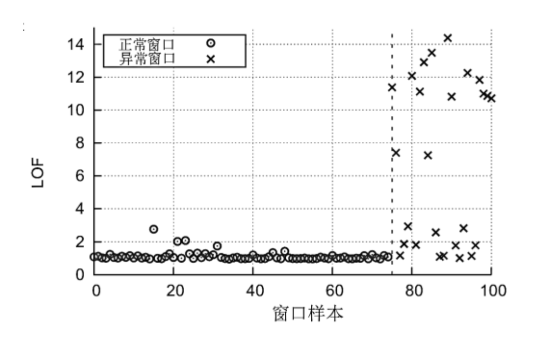
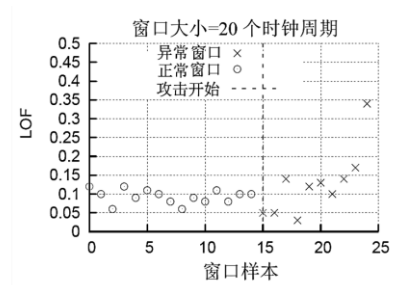


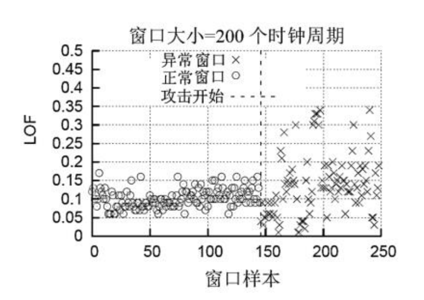
图 11　Downloader 上的 LOF 结果分布

5.4　性能分析

检测延迟是指从攻击发生到 CVSM 检测到异常之间的时间，从该角度出发对系统性能进行分析。通过对延迟时间进行测试不仅可以评估算法性能，还能评估 CVSM 的实时性。

窗口大小的设定会对检测延迟产生影响，较长的窗口可以捕获到更多系统行为，为学习算法提供更充分的分类信息，但同时也产生了较大的计算量增加了检测延迟。设置不同窗口大小时正常行为和异常行为的 LOF 数值分布如图 12 所示，其中时钟周期为 10 毫秒。





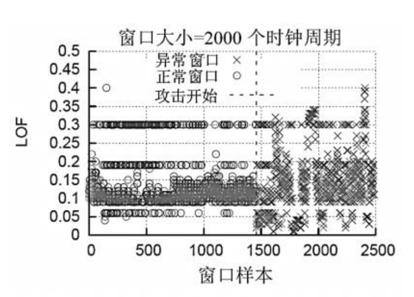


图 12　窗口大小与样本 LOF 值分布

由图 12 可以看出，当设置较小的窗口（20 个时钟周期）时所含样本较少，难以形成有效的密度统计，即使在注入恶意

程序之前也有可能随机出现 LOF 值偏高的正常样本，这些点可能会被误判为异常，导致整个窗口被误分为异常窗口；当设置较大的窗口（2 000 个时钟周期）时包含样本较多，样本间的相对密度较大，正常样本和异常样本的 LOF 值都偏高而难以识别出其中的异常点，容易导致整个窗口被误分为正常窗口。测试表明当设置窗口大小为 200 个时钟周期（2 s）时可以取得较好的分类效果，该时间段内可以向后端提供充足的分类信息，也能够在数秒钟的延迟内检测出异常行为。

在确定窗口大小后测试了 CVMS 的检测延迟，以 Database 负载为例的结果如图 13 所示，X 轴表示从注入恶意程序到生成警报之间的时间延迟，Y 轴表示时间延迟内的检测率。

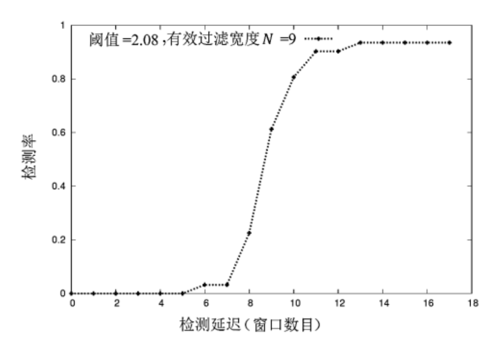


图 13　LOF 算法在 Database 负载上的检测延迟

检测率和检测时间之间存在一定关系：a)较长的检测时间可以对多个窗口进行持续观测，积累较多的事件特征值，有利于提高检测率。但如果等完全确认某操作存在威胁时才生成警报，则该操作可能已经对系统造成破坏;b)如果检测时间过短而提前生成警报，则可能产生较多的误报现象。因此必须在保持适度检测率和误报率的前提下及时生成警报。

表 10　不同负载上的检测率与检测延迟

	LOF			KNN		
负载	TP(%)	FP(%)	检测延迟	TP(%)	FP(%)	检测延迟
Database	98	6	11	98	6	9
Web	96	0	1	97	13	1
EMail	84	3	14	94	3	15

恶意程序注入到正常负载中后可能并不立即产生异常行为，则从注入到表现出异常之间存在潜伏期，这种情况会增加 CVSM 的实际检测延迟，因此考虑了多数（95%）不含潜伏期恶意程序时的检测延迟。表 10 列出了不同负载上的检测率、误报率和对 95%样本的检测延迟。从结果中可以看出，LOF 和 K-NN 算法在 Web 负载上具有较高的检测率，同时也具有 1 个窗口的最短检测延迟（2 s），而两种算法在 Email 负载上的检测延迟最长（约 30 s）。总体上，CVSM 可以在平均 8.5 秒的延迟内以较高的检测率和中等误报率识别出异常行为。

6　结束语

本文针对虚拟机语义鸿沟问题，根据硬件体系结构数据难以被窜改和旁绕的特性收集了 VMM 层捕获的数据，增强了数据源的可靠性。通过对 CVSM 的适应性设计，使之满足能够进行数据挖掘的必要条件，构造了速率特征和关系特征，为机器学习过程提供了准确而简洁的数据，建立了基于特征选择的语义鸿沟桥接模型。实验结果表明，CVSM 仅利用 VMM 层收集的数据便能够完成异常检测，引入机器学习方法处理虚拟机数据能够有效减小虚拟机特征空间，建立起有效的虚拟机执行模式并识别出客户机的异常行为，提高了语义鸿沟桥接效率，为处理语义鸿沟问题提供了一种可行方案。

参考文献：

[1] Garfinkel T, Rosenblum M. A virtual machine introspection based architecture for intrusion detection [C]// Proc of the 10th Annual Network and Distributed System Security Symposium. California: CPlane Inc, 2003: 191-206.

[2] Schneider C, Pfoh J, Eckert C. A universal semantic bridge for virtual machine introspection [C]// Proc of the 7th International Conference on Information Systems Security. Berlin: Springer, 2011: 370-373.

[3] Ibrahim A S, Hamlynharris J, Grundy J, et al. Supporting virtualization ware security solutions using a systematic approach to overcome the semantic gap [C]// Proc of the 5th International Conference on Cloud Computing. Washington DC: IEEE Computer Society, 2012: 836-843.

[4] Hizver J, Chiueh T C. Real-time deep virtual machine introspection and its applications [C]// Proc of the 10th ACM SIGPLAN//SIGOPS International Conference on Virtual Execution Environments. New York: ACM Press, 2014: 3-14.

[5] Harrison C. Odinn: an in-vivo hypervisor-based intrusion detection system for the cloud [J]. IEEE Trans on Power Delivery, 2014, 12 (1): 354-363.

[6] 林杰，刘川意，方滨兴. IVirt: 基于虚拟机自省的运行环境完整性度量机制 [J]. 计算机学报, 2015, 38 (1): 191-203.

[7] Fu Yangchun. Bridging the semantic gap in virtual machine introspection via binary code reuse [D]. Dallas: The University of Texas, 2016.

[8] Kannan A, Maguire G Q, Sharma A, et al. Genetic algorithm based feature selection algorithm for effective intrusion detection in cloud networks [C]// Proc of the 12th International Conference on Data Mining Workshops. Washington DC: IEEE Computer Society, 2012: 416-423.

[9] Tupakula U, Varadharajan V, Dutta D. Intrusion detection techniques for virtual domains [C]// Proc of the 19th International Conference on High Performance Computing. Washington DC: IEEE Computer Society, 2013: 1-9.

[10] Xu Zhixing, Ray S, Subramanyan P, et al. Malware detection using machine learning based analysis of virtual memory access patterns [C]// Proc of Design, Automation and Test in Europe Conference and Exhibition. Washington DC: IEEE Computer Society, 2017: 169-174.

[11] Mishra P, Pilli E S, Varadharajan V, et al. Securing virtual machines from anomalies using program-behavior analysis in cloud environment [C]// Proc of the 18th International Conference on High Performance Computing and

Communications. Washington DC: IEEE Computer Society, 2016: 991-998.

[12] 娄睿. 虚拟多域环境的安全保护技术研究 [D]. 郑州：信息工程大学, 2014.

[13] More A, Tapaswi S. Virtual machine introspection: towards bridging the semantic gap [J]. Journal of Cloud Computing, 2014, 3 (1): 16-30.

[14] Tsuchiya M, Fujiyoshi H. A method of feature selection using contribution ratio based on boosting [C]// Proc of the 19th International Conference on Pattern Recognition. Washington DC: IEEE Computer Society, 2008: 1-4.

[15] Bijalwan V, Kumar V, Kumari P, et al. KNN based machine learning approach for text and document mining [J]. International Journal of Database Theory and Application, 2014, 7 (1): 61-70.

[16] Bhatt V, Dhakar M, Chaurasia B K. Filtered clustering based on local outlier factor in data mining [J]. International Journal of Database Theory and Application, 2016, 9 (5): 275-282.